CARPO: Correlation-Aware Power Optimization in Data Center Networks

Xiaodong Wang and Xiaorui Wang

Department of Electrical and Computer Engineering

The Ohio State University

{*wangxi, xwang*}*@ece.osu.edu*

Abstract. Power optimization has become a key challenge in the design of large-scale enterprise data centers. Existing research efforts focus mainly on computer servers to lower their energy consumption, while only few studies have tried to address the energy consumption of data center networks (DCNs), which can account for 20% of the total energy consumption of a data center. In this paper, we propose CARPO, a correlation-aware power optimization algorithm that dynamically consolidates traffic flows onto a small set of links and switches in a DCN and then shuts down unused network devices for energy savings. In sharp contrast to existing work, CARPO is designed based on a key observation from the analysis of real DCN traces that the bandwidth demands of different flows do not peak at exactly the same time. As a result, if the correlations among flows are considered in consolidation, more energy savings can be achieved. In addition, CARPO integrates traffic consolidation with link rate adaptation for maximized energy savings. We implement CARPO on a hardware testbed composed of 10 virtual switches configured with a production 48-port OpenFlow switch and 8 servers. Our empirical results with Wikipedia traces demonstrate that CARPO can save up to 46% of network energy for a DCN, while having only negligible delay increases. CARPO also outperforms two state-of-the-art baselines by 19.6% and 95% on energy savings, respectively. Our simulation results with 61 flows also show the superior energy efficiency of CARPO over the baselines.

1 Introduction

In recent years, high energy consumption has become one of the most important concerns for large-scale data centers that are rapidly increasing the number of hosted servers. For example, in a 2007 report to the US Congress [1], the Environmental Protection Agency (EPA) estimated that the annual data center energy consumption in the US will grow to over 100 billion kWh at a cost of $7.4 billion by 2011. As a result, minimizing the energy consumption of data centers has recently attracted a lot of research efforts. However, current research focuses mostly on computer servers to lower their energy consumption, while only few studies have tried to address the energy consumption of data center networks (DCNs), which can account for 10% to 20% of the total energy consumption of a data center [2][3]. The percentage of network energy can even increase to 50% in future data centers, where servers become more energy-proportional to their workloads [4]. Similar to servers, the networks in data centers are commonly provisioned for the worst-case workloads that rarely occur. As a result, the capacity of a DCN is usually far from being fully utilized. Therefore, significant power[1] savings can be achieved by making DCNs more energy-proportional to their workloads as well.

It is well known that the power consumption of a network is mostly independent of its workload, mainly because the components on network devices, such as switch chips and fans, consume a significant amount of power even with low workloads [3][5]. There are some existing energy-efficient designs for traditional local and wide area networks. For example, some studies have proposed to put network devices, such as switches and routers, into sleep during periods of very low traffic activities, and buffer the packets during the sleeping period to transmit at a later time [6][7]. While this approach works effectively for traditional networks, it is less applicable to today's typical multi-tiered DCNs, because their high degrees of oversubscription can lead to much shorter idle periods [4]. As a result, frequently putting network devices into sleep intermittently in a DCN may cause packets to be buffered or rerouted around the deactivated switches and links. Buffering

[1] Similar to related work [3][5], we use power and energy interchangeably in this paper, because data center networks are typically required to be always-on. Also, we do not address heat density or electricity costs in this paper.

packets in a DCN for the purpose of energy savings may cause packets to be dropped or backpressure depending on the adopted flow control mechanism [4]. Frequent routing changes may lead to considerable overheads for coordination.

To achieve energy-proportional DCNs, two recent studies proposed energy-efficient approaches that are more amenable to DCNs. First, in their work called *ElasticTree*, Heller et al. [3] studied real traces from a production data center and demonstrated that traffic flows in a DCN can be consolidated onto a small set of links and switches, which are sufficient to serve the bandwidth demands for most of the time. Second, Abts et al. [4] proposed a link rate adaptation approach that dynamically estimates the future bandwidth needs of each link and then reconfigures its data rate to achieve power savings.

While traffic consolidation has been demonstrated to be a highly effective way to achieve energy proportionality in DCNs by shutting down unused network devices, existing work consolidates traffic flows in a greedy way and assumes that the bandwidth demand of each data flow can be approximated as a constant during the consolidation process. This is in contrast to the fact that the bandwidth demand of a traffic flow can vary over time. The variations can be significant because the consolidation period normally cannot be very short due to overhead considerations [8]. Therefore, existing work has to use either estimated maximum or average demands to perform consolidation, which can result in either unnecessarily high power consumption or undesired link capacity violations, respectively. A key observation based on the analysis of real DCN traces is that the bandwidth demands of different flows usually do not peak at exactly the same time. As a result, if the correlations among flows are considered in consolidation, more power savings can be achieved. Another important observation is that the 90-percentile bandwidth demands are usually half or less of the peak demands. Therefore, if we could avoid consolidating traffic flows that are positively correlated (*e.g.*, peak at the same time) based on 90-percentile demands instead of peak demands, we may further improve the energy efficiency of traffic consolidation. To our best knowledge, this work presents the first study that applies correlation analysis to traffic consolidation in DCNs.

In this paper, we propose CARPO, a correlation-aware power optimization scheme that consolidates traffic flows based on correlation analysis among flows in a DCN. Another important feature of CARPO is to integrate correlation-aware traffic consolidation with link rate adaptation for maximized energy savings. The integration is formulated as an optimal flow assignment problem, which is known to be NP-Complete. A near-optimal solution is first computed using a linear programming tool to determine consolidation and the data rate of each link in the DCN. To reduce the computation complexity, we then propose a heuristic algorithm to find a consolidation and rate configuration solution with acceptable runtime overheads. We then discuss the practical considerations of implementing CARPO in real data centers and present an analytical model that theoretically estimates the energy savings of CARPO without implementing it in a DCN. As a result, a DCN administrator can use the estimation to compare the energy savings of different network topologies and choose one that can provide the desired trade-offs between power and performance. Specifically, the major contributions of this paper are as follows:

- By analyzing real data center traces, we observe that DCN network flows usually have weak pairwise correlations and thus do not peak at the same time. Based on the observation, we propose to consolidate traffic flows according to their correlations, such that significant power savings can be achieved.
- To further improve the DCN energy efficiency, we propose to integrate link rate adaptation, an existing energy saving approach amenable to DCNs, with the correlation-aware traffic consolidation approach. The integration is formulated as an optimal flow assignment problem.
- To solve the optimal flow assignment problem, we propose a heuristic algorithm to find the consolidation and rate configuration solution with acceptable overheads. Compared with the solution provided by formal mathematical programming software, our algorithm has only negligible performance degradation, but significant computational efficiency improvement.
- We implement our CARPO solution both on a small scale hardware testbed and in a larger scale network simulation. Both the empirical experiments and simulation

running real data center traces demonstrate that our CARPO solution outperforms two state-of-the-art baselines on power savings with only negligible delay increases.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 formulates the integration of traffic consolidation and link rate adaptation. Section 4 analyzes the correlations of traffic flows from the Wikipedia traces and presents the CARPO framework and the proposed correlation-aware algorithm. Section 5 introduces the implementation of our hardware testbed and the evaluation results. Section 6 presents the simulation results. Section 7 concludes the paper.

2 Related Work

There have been some studies on the power consumption of network equipment. Gupta et al. [9] have explored the possibility of putting network devices into sleep, such as switches and routers, when the traffic activities are low, to save the network device power. They also proposed an abstract sleep model in [6], which is used to calculate the power savings by putting the network devices into sleep. Ananthanarayanan et al. [10] used a time prediction window to predict the packet coming time, such that the network device can be shut down when there is no packets coming. However, the aforementioned power-saving schemes are not designed for DCNs, which have special traffic characteristics. The oversubscription of a multi-tiered DCN leads to short idle periods [4] of the network, which may result in a high transitional power consumption on the network device because of periodic sleeping. In contrast to the previous projects, we focus on DCNs to find power-saving strategies that do not need to *frequently* put network devices into sleep and wake them up based on transient traffic activities.

DCN research has recently received a lot of attention. For example, novel DCN architectures have been proposed in [11][12][13], but those projects do not address DCN power consumption. Several recent studies have proposed different approaches to achieve energy-proportional DCNs. The first approach is *traffic consolidation* proposed in *ElasticTree* [3]. The researchers of the *ElasticTree* project propose to consolidate traffic flows in a DCN onto a small set of links and switches such that unused network equipments

can be turned off for power saving. Although their consolidation approach is developed based on a real data center traces, they assume that the traffic rate of each data flow is approximately a constant, which may not be valid for many of the current production data centers due to their high variations in workloads. In contrast to the consolidation methods in the *ElasticTree*, we analyze the statistical characteristics of data center traffic flows based on the real data center traces. The statistical analysis provides a guidance for us to propose correlation-aware traffic consolidation, which is shown to have more energy savings. The second approach to energy-proportional DCN is to adapt the link rate according to the workload of each traffic flow [7][4][14]. One problem of solely using link rate adaptation is that the power saving gain is relatively small because the majority amount of power is consumed by components such as fans and switch fabrics. In this paper, we propose to integrate link rate adaptation with correlation-aware traffic consolidation, such that more power savings can be achieved.

3 Problem Formulation

In this section, we first analyze the power characteristics of network switches. Based on the analysis, we formulate the integration of traffic consolidation and link rate adaptation as a constrained optimization problem.

3.1 Characteristics of Network Switch

A general-purpose network switch is commonly composed of chassis, switching fabric, line-cards, and ports. Switch chassis includes cooling equipments, such as fans, which usually consumes a fixed amount of power. The switching fabric maintains the switching table while the line-card buffers all the incoming and outgoing packets. Ports contain the networking circuitry, which consumes different amount of power at different speed. According to the measurements from [15], the idle power consumption of a 48-port edge LAN switch ranges from 76W to 150W. Around 40 Watts or more can be added if the switch is working under the maximum capacity. These measurements indicate that each switch port only consumes 1-2 Watts, while the switch chassis, fabric, and line-cards

consume most of the power. Therefore, compared with strategies that only fine tune the transmission rate of each port, more significant power savings can be achieved if unnecessary switches can be turned off.

On the other hand, Benson et al. studied the data center link utilization characteristics in [16] for real production data centers and found that the average link utilization of the aggregation layer links is only about 8% during 95% of the time, while the average link utilizations of the edge layer links and the core layer links are approximately 20% and 40%, respectively. The low link utilizations provide us a large space to consolidate traffic flows from different links, such that a smaller number of links and switches are needed to provide services to all the existing traffic workloads.

To verify the power consumption characteristics of the switch, we measure the power consumption on a production PRONTO 3240 OpenFlow-enabled switch. We tune the port speed settings to different levels (inactive/10/100/1000Mbps) in different rounds of measurement and measure the total power consumption of the switch. The measurement results are listed in Table 1. We can see that the switch itself with no active port consumes more than half of the total power when all ports are idling with the highest speed setting. If the switch is working under 100% utilization at all ports, 5% more power consumption is to be added to the total power [15]. Note that power consumption varies about 10-20 Watts between each level of port speed.

Table 1. Power consumption measurements of a 48-port PRONTO 3240 switch used in our testbed.

PRONTO 3240	Port Speed (Mbps)			
Switch	None Active	10	100	1000
Power (W)	67.7	70.7	80.2	111.5

3.2 Optimal Consolidation and Rate Configuration

We now formulate the flow consolidation problem as an optimal flow assignment problem. We assume that the flows we need to consolidate cannot be split, such that little impact is introduced to connection-oriented flows, such as TCP flows. The optimization goal is to assign traffic flows to network links for every computational period, such that the traffic

constraints are satisfied and the energy consumption during that period is minimized. This problem is an NP-complete problem for integer flows [3]. The overall idea of our design is to select the minimum subset of network devices, which can provide enough capacity for all traffic flows in the network.

We first define the following notation:

- P^{switch}: the power of one switch, including power consumed by both the chassis, the line-cards and the ports;
- $P^{chassis}$: the power consumed by a single switch chassis;
- P^{cards}: the power consumed by all line-cards on a switch;
- P^{port}: the power consumed by each port, which includes both the static power P^{port_s} and the dynamic power P^{port_d};
- P^{port_f}: the dynamic power of a port in full link capacity;
- din^j: the data rate of the incoming flow at port j;
- $dout^j$: the data rate of outgoing flow at port j;
- C: the maximum link capacity.

Assume we have N switches, each switch has p ports, and the consolidation period is T, the problem can be formulated as follows:

$$\min \sum_{t=0}^{T} \sum_{i=0}^{N} P_{i,t}^{switch} \tag{1}$$

where

$$P_{i,t}^{switch} = P_{i,t}^{chassis} + P_{i,t}^{cards} + \sum_{j=0}^{p} (P_{i,t,j}^{port_s} + P_{i,t,j}^{port_d}) \tag{2}$$

$$P_{i,t,j}^{port_d} = P_{i,t,j}^{port_f} \times \frac{(din_{i,t}^{j} + dout_{i,t}^{j})}{C} \tag{3}$$

under the following constraints:

- For each time point, the link capacity is not exceeded

$$din_{i,t}^{j} \leq C \quad dout_{i,t}^{j} \leq C \quad \forall i, j, t; \tag{4}$$

– For each switch, the number of packets that are received equals to the number of packets that are sent out

$$\sum_{t=0}^{T}\sum_{j=0}^{p} din_{i,t}^{j} = \sum_{t=0}^{T}\sum_{j=0}^{p} dout_{i,t}^{j} \quad \forall i; \tag{5}$$

– All traffic flows are assigned to paths in the network.

To mathematically solve the above optimization problem, we use MathProg, a mathematical programming language, with the GLPK (GNU Linear Programming Kit) mathematical programming software. The optimization model is transformed to a mixed-integer programming (MIP) model for programming purpose. The usage of each link is defined as a binary variable. If a link is used, the corresponding binary variable is 1; otherwise it is 0. The status of each switch is also represented by a binary variable, where 1 indicates the switch is turned on while 0 means it is turned off.

While solving the problem with the above MIP and GLPK tools can give us a near-optimal solution for energy savings, it cannot be used at runtime in practice because 1) it assumes perfect knowledge of future bandwidth demands of the traffic flows by reading the information from the DCN traces, and 2) its computation complexity is too high. The algorithm used in the above programming softwares includes two major steps, a Simplex algorithm step and an integer programming step. Since the complexity of the second step has been proven to be exponential [17], the complexity of the entire algorithm is thus exponential.

4 Design of CARPO

Due to the lack of perfect knowledge about the bandwidth demands of the traffic flows in a DCN in practice, we now propose CARPO to estimate the correlations among different flows in traffic consolidation. In this paper, similar to related work [3][4][5], we assume a centralized power manager, which allocates a path for each traffic flow, adapts the link rate for each port, and turns off unused switches and ports. In very large DCNs, CARPO can be extended to work in a distributed way, which is our future work.

4.1 Traffic Correlation Analysis

To motivate the design methodology of CARPO in later sections, we now present our analysis on traffic traces from Wikipedia's data centers, which consist of 10% of 4 months of actual HTTP requests [18]. To simplify our analysis, we choose the traces from the last 7 days of all the available traces for analysis. We focus on analyzing the response traffic flows sent for all the HTTP requests. The response traffic flows can be generated from the requests. We assume that for each request, a file is sent back to the users from a corresponding database server in the DCN. In that case, the response traffic flows can be converted to traffic flows between the end server and the database server within the DCN. To generate a reasonable number of traffic flows, we group the response files together based on their folder names, such that all files from the folders with the same first character (case sensitive) in their names are considered to locate on the same database server. By following this folder-based grouping rule, we finally get 61 database servers, which leads to 61 traffic flows, assuming one database server is paired with one end host. To generate the actual response traffic workload, we assign each response file a size following the distribution introduced in [19]. Figure 1 shows 4 example flows from the final 61 flows. We can see that the traffic flows have high variations within itself, but they usually do not peak together.

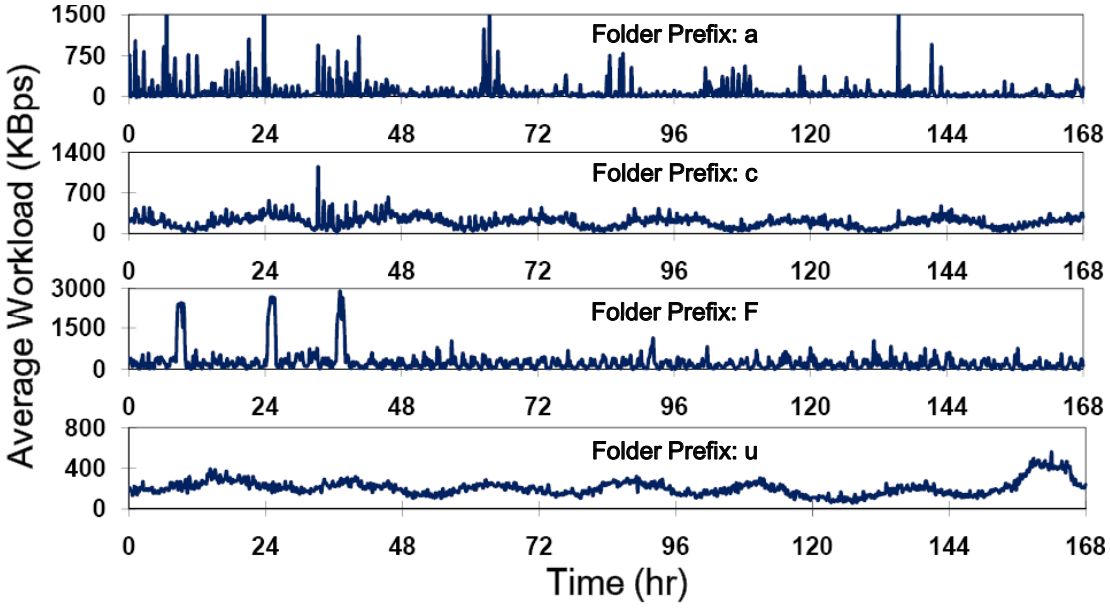


Fig. 1. Four example traffic flows from the aggregated Wikipedia traces. The duration of each trace is seven days with one data point each second.

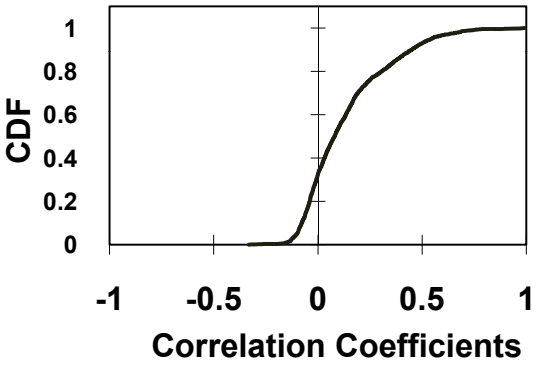


Fig. 2. CDF distributions of pair-wise correlation coefficients in the Wikipedia traces.

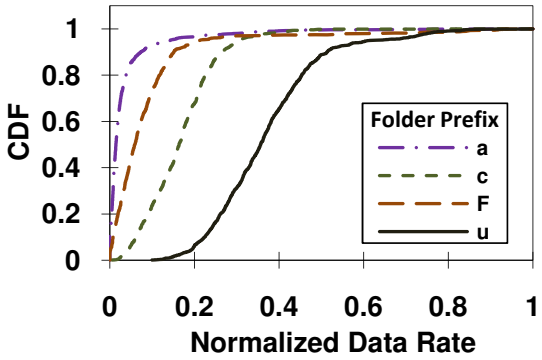


Fig. 3. CDF of the normalized data rate for the flows in Figure 1. The data rate of each flow is normalized to its own peak value.

It can also be observed that some flows show negative correlation, such as the flows with folder prefix as c and u. In this case, the two flows have approximately opposite variation trends. To quantify the correlation relationship of each flow pair, we calculate the Pearson Correlation Coefficient between each flow pair by

$$r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}} \tag{6}$$

where x_i and y_i are the bandwidth demands of flows x and y at each sample point within one consolidation period.

The CDF distribution of all the pair-wise correlation coefficients in the Wikipedia traces is shown in Figure 2. We see that more than 80% of the traffic flows are loosely correlated or even negatively correlated together with a correlation value of less than 0.3. This distribution leads to our **Observation 1:** *traffic flows within a data center are usually loosely correlated together, and so usually do not peak at the same time.*

We then study the statistical properties of traffic flows in the Wikipedia data centers. Figure 3 shows the CDF distribution of the normalized data rate of each traffic flow in Figure 1. The data rate is normalized to each flow's own maximum bandwidth demand. From Figure 3 we see that all the four flows have a 90-percentile normalized data rate less than 50% of their own peak bandwidth. This is consistent with the analysis of data

center traffic characteristics in [16] that the 95-percentile of the link utilization is less than 10% in a real production data center. Based on these statstical data and the analysis in the previous work, we can have **Observation 2:** *the 90-percentile link utilizations for most flows are usually much less than their own peaks, which suggests that if traffic flow consolidation is based on the off-peak values (e.g. 90-percentile), more power savings can be achieved.*

Based on the previous two observations, we design CARPO, a correlation-aware power optimization scheme for the DCN. In general, CARPO periodically calculates the correlations among flows and consolidates traffic flows based on the 90-percentile of each traffic flow's peak demand, such that only a small set of the network devices is needed to hold all the traffic flows while the rest of the network devices can be shut down for power savings. The detailed design of CARPO is presented in later sections. The key novelty of CARPO is the adoption of correlation analysis for improved traffic consolidation.

4.2 CARPO Framework

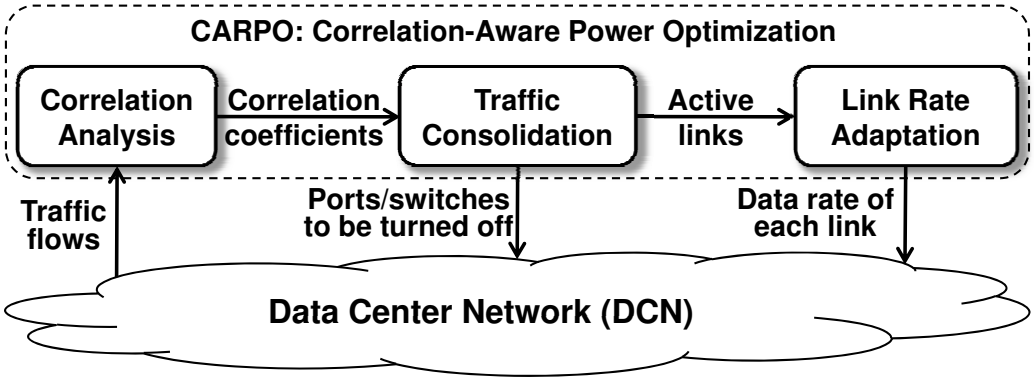


Fig. 4. The proposed CARPO framework.

In general, as shown in Figure 4, CARPO takes three steps, *Correlation Analysis*, *Traffic Consolidation* and *Link Rate Adaptation* to optimize the power consumption of a DCN. In the first step, CARPO takes the data rates of the traffic flows in the previous consolidation periods as input and analyzes the correlation relationship between different traffic flows using the method introduced in Section 4.1. In the second step, based on the correlation coefficients from the previous analysis, CARPO uses the 90-percentile

data rate of each link in the previous period to consolidate the traffics under the link capacity constraint. After the consolidation, unused switches and ports are turned off for power savings. In the last step, CARPO adapts the data rate of each active link to the demand of the consolidated traffic flows on that link, such that more power savings can be achieved for the DCN.

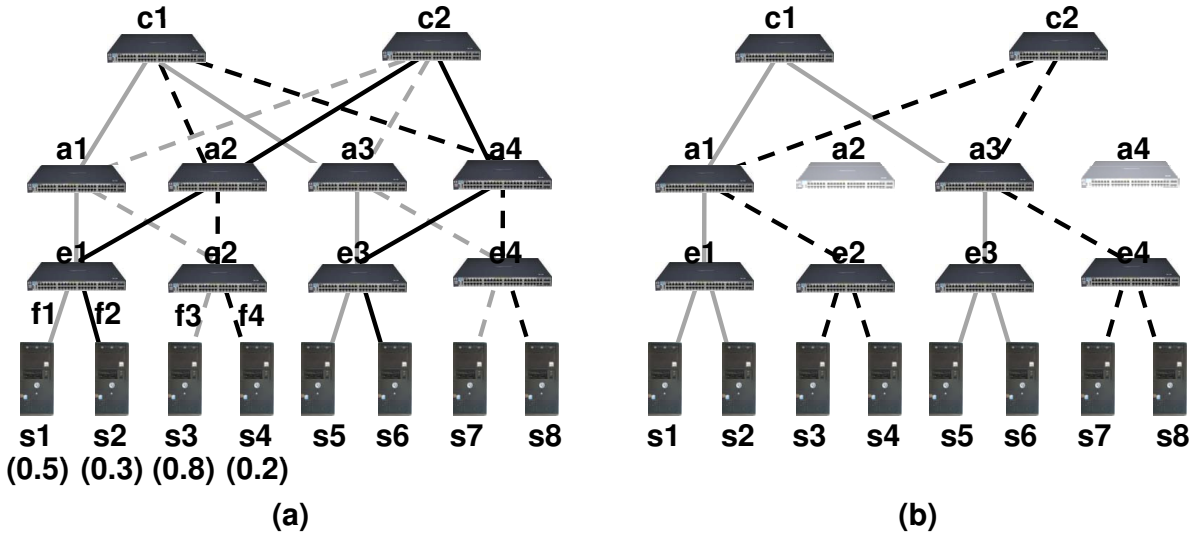


Fig. 5. An example of traffic consolidation by CARPO. (a) shows the initial flow assignment. (b) shows the aggregated flow assignment after applying CARPO. s_i is the ith server. f_j is jth flow. e_i, a_i, c_i represent the *edge*, *aggregation* and *core* switches, respectively. Links with different color-pattern combinations represent flow assignments. The 90-percentile bandwidth demands of the four flows are listed below their source servers in (a).

Table 2. Example correlation values for flow pairs in Figure 5.

Flow Pairs	f_1,f_2	f_1,f_3	f_1,f_4	f_2,f_3	f_2,f_4	f_3,f_4
Correlation	-1	-1	1	-1	-1	-1

Figure 5 presents a simple example to illustrate how CARPO works. In this example, there are four traffic flows: $f_1 : s1 \rightarrow s5$, $f_2 : s2 \rightarrow s6$, $f_3 : s3 \rightarrow s7$ and $f_4 : s4 \rightarrow s8$. The correlation value between each flow pair is listed in Table 2. The normalized 90-percentile capacity demand of each flow is labeled on the flow's source server in Figure 5(a). Figure 5(a) shows an initial flow placement setting of the network. Figure 5(b) is the traffic consolidation result after we apply CARPO on the initial flow setting. More specifically,

flow f_1 and f_2 are aggregated together since they have low correlation (-1) and their aggregated 90-percentile data rate does not violate the link capacity. Flow f_3 cannot be aggregated with f_1 and f_2 because the aggregated 90-percentile data rate violates the capacity constraint of the link between switch a_1 and c_1. Although the total bandwidth demand of flows f_1, f_2, and f_4 does not violate the capacity constraint, f_4 should not be aggregated with f_1 and f_2 because it has a high correlation (+1) with f_1, which means that f_1 and f_4 are going to peak together, resulting in a high probability of link capacity violation. Finally, since f_3 and f_4 have negative correlation and their aggregated 90-percentile data rate does not violate the capacity constraint, they should be aggregated together. After consolidating all the flows, switches $a1$ and $a4$ can be turned off to save power, since they do not serve any flows. In the last step, all port speeds are set to the maximum requirement of the each aggregated data link, rather than the maximum available speed, in the new consolidated network, such that more power can be saved with lower data rates.

4.3 Correlation-Aware Consolidation Algorithm

We have introduced in Section 3.2 that using the mathematical tools to solve the consolidation and rate adaptation problem is infeasible mainly because of the high computational complexity. We now propose a light weight heuristic algorithm that achieves the correlation-aware consolidation with low computational overhead. We assume that the link in our DCN is duplex with same capacity for upstream and downstream flow. The algorithm we designed is based on the greedy-bin packing algorithm. We greedily assign as many traffic flows as possible to a single path. The pseudo code of our algorithm is presented in Algorithm 1. The algorithm takes the flow list F, the link list l, link capacity c, path link list $PATHL$ for each available path, and the correlation threshold Cor_{th} as input. Note that each entry in $PATHL$ is a link list of one path. The path list is ordered from left to right based on the network topology. In Algorithm 1, Lines 1-2 initialize the correlation value (Cor) between all flow pairs, and the data rate ($rate$) of each flow. Lines 3-16 assign each flow to a path. More specifically, Line 6 calculates if adding the current

flow f_i to *path* j would violate the capacity of any available link on *path* j. Line 7 checks if the correlation between flow f_i and the flows existing on *path* j meets the correlation requirements. If both of these two requirements are satisfied, flow f_i is assigned to *path* j and the available link capacity of each link along *path* j is updated (Lines 8-10). Note that the available link capacity is updated with the 90-percentile link utilization value of the aggregated traffic after the new flow is assigned in each step. Program terminates when all the flows are assigned.

The complexity of the algorithm is determined by the number of switches in the network and the number of flows that the network needs to serve. If the network has N switches and serves M traffic flows, the worst-case number of paths that a flow can take is in the order of $O(N^2)$ when the core switches and the aggregation switches are fully connected. Therefore, the complexity of the algorithm is $O(MN^2)$;

Algorithm 1 Correlation-aware Traffic Consolidation

Require: Flow list $F = \cup\{f_i\}$ with M flows, correlation threshold Cor_{th}, link list l, link capacity c and link list $PATHL$ for each available path.
Ensure: Final path of each flow $PATHF$

1: $Cor[M][M] = CORRELATION(F)$;
2: $rate[M] = Ninety\_Percentile(F)$;
3: **while** $F \neq NULL$ **do**
4: **for** $j = 1$ to m **do**
5: **for** $\forall f_i$ that can take path j **do**
6: **if** $c[k] - rate[i] \geq 0\ \forall l[k] \subset PATHL[j]$ **then**
7: **if** $Cor[i][i'] \leq Cor_{th}\ \forall f_{i'} \subset PATHF[j]$ **then**
8: $PATHF[j] = PATHF[j] \cup \{f_i\}$;
9: $F = F - \{f_i\}$;
10: $c[k] = UPDATE(PATHL[j], rate[i])\ (\forall l[k] \subset PATHL[j])$;
11: **end if**
12: **end if**
13: **end for**
14: **end for**
15: **end while**
16: **return** $PATHF$

4.4 Link Rate Adaptation

We have seen that correlation-aware traffic consolidation provides us an efficient way to save DCN power. The algorithm we designed is essentially based on the bin-packing

algorithm. As a result, links may not be fully utilized after flows are consolidated. We further propose to adapt the data rate of each link to the consolidated traffic data rate, such that the link power consumption can be reduced, leading to more overall power savings. The switches used by DCNs are commonly manufactured with different tunable link rates [4]. In our work, we assume the switch is a commodity switch with link rates available at 10Mbps, 100Mbps and 1Gbps, as shown in Table 1 in Section 3.1. The link rate adaptation is performed on a much finer granularity, with a frequency of every second.

5 Hardware Evaluation

In this section, we first introduce the hardware testbed used for our experimental evaluation. We then present the evaluation results from our testbed running the Wikipedia workload.

5.1 Hardware Testbed and Baselines

We build our hardware testbed with 8 servers serving 4 flows, a 48-port Pronto 3240 OpenFlow-enabled switch, and a desktop as the flow manager. The topology of the network in the testbed is the same as the one shown in Figure 5. Each server is equipped with two AMD Athlon(tm) 64 X2 Dual Core processors with 4GB RAM. To form the 10 switches in our testbed, we configure the Pronto switch to have 10 virtual switches, each with 4 ports as shown in Figure 6. The flow manager desktop is connected to the Pronto switch via a RJ45 to DB9 console cable, serving as a centralized manager. The flow manager calculates the new flow assignments every 10 minutes and sends the new path configuration to the switch. The switch then updates the internal flow table and directs each flow accordingly.

To evaluate the performance of the proposed *CARPO* strategy, we compare *CARPO* with the following baseline schemes:

- *ElasticTree* [3] is a state-of-the-art traffic consolidation solution. In contrast to CARPO, *ElasticTree* assumes that all traffic flows are constant bit rate traffic in one consolidation period. In the experiment, we use the the data rate of the last sample point

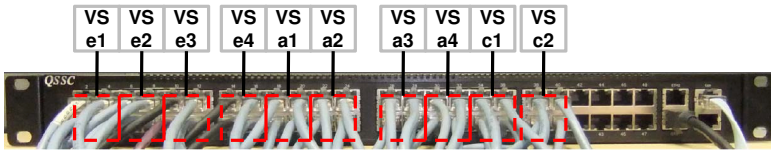


Fig. 6. Hardware testbed with 10 virtual switches (VSs) configured from a production 48-port OpenFlow switch and 8 servers. The VSs are numbered in the same way as in Figure 5.

in the last period as the traffic bit rate. The key differences between *ElasticTree* and *CARPO* are that: 1) *CARPO* consolidates different traffic flows based on the correlation analysis and 90-percentile properties of each flow; 2) *CARPO* integrates link rate adaptation for maximized power savings.

– *GoogleP* [4] is the second baseline. *GoogleP* predicts the future bandwidth demand of each link, and then tunes the data rate of each link to the lowest level that can meet the bandwidth demand.

To highlight the improvement resulting from correlation-aware consolidation, we also present the results of *CARPO-C*, a variant of *CARPO* that only conducts correlation-aware consolidation without link rate adaptation. To investigate the solution quality of *CARPO*, we also present the results of the *Near-optimal* solution, which is the oracle solution to the formal model presented in Section 3.2. Note that *Near-optimal* is impractical because 1) it assumes perfect knowledge of future bandwidth demands of the traffic flows by reading the information from the DCN traces ahead of the time, and 2) its computation complexity is high. Actually, due to its high complexity, we can only present the results of *Near-optimal* for the small topology used in our testbed experiments, but not for the larger topology used in the simulations in later sections.

5.2 Testbed Results

The 4 traffic flows used in the experiment are the ones in Figure 1. Each flow is served by one pair of servers, leading to 8 servers in total. The sources of the four flows are the 4 servers located at the left side of the topology as shown Figure 5, while the other 4 servers serve as the destinations.

The performance metrics we evaluate are energy savings that each power management scheme can achieve over the case when there is no power management in the DCN. According to the CARPO framework, the flow configuration in the DCN needs to be updated every 10 minutes and the unused switches and ports should be shut down. However, since our testbed is established with 10 virtual switches on a single physical switch, each individual virtual switch cannot be shut down separately. Therefore, to quantify the power consumption of each virtual switch, we measure the power consumption of the entire physical switch with no ports turned on, which is 66.7 W, leading to 6.67W for each virtual switch as its chassis power consumption. We also measure the power consumption of a single active port at different data rates, which are 1W at 1 Gbps, 0.3 W at 100 Mbps and 0.15 W at 10 Mbps. By having these measurements, we can obtain the empirical power consumption model of the entire network in our testbed by:

$$P = 6.67 \cdot N_s + \sum_{i=1}^{3} (P_i \cdot J_i) \tag{7}$$

where N_s is the number of active virtual switches, P_i is the active power of a single port at the data rate level i and J_i is the corresponding number of active ports at that data rate level. For *CARPO-C* and *ElasticTree*, the second item in Equation (7) is actually $1 \cdot J_p$, where J_p is the number of active ports, since all ports have the same data rate of 1Gbps without rate adaptation. For *GoogleP*, the power consumption is $66.7 + \sum_{i=1}^{3} (P_i \cdot J_i)$ since no virtual switch can be turned off by *GoogleP* at any time.

The power consumption of *CARPO* and the baselines for the first day of the 7-day Wikipedia traces is shown in Figure 7 in a fine granularity. We can see that *GoogleP* consumes the most power because it only adapts the link rates in response to workload variations for power savings. Since majority of the DCN power is consumed by the switch chassis, *GoogleP* is not capable of saving that portion of power. *ElasticTree* has a higher power consumption, compared with *CARPO* and *CARPO-C*, because it only uses the last sample point from the last consolidation period as a constant bit rate for the next consolidation period. Different from *ElasticTree*, *CARPO-C* considers the correlation and 90-percentile bandwidth demands of the traffic flows in traffic consolidation, leading to

more power savings, as discussed in Section 4. *CARPO* has the lowest power consumption because it integrates correlation-aware consolidation with link rate adaptation. The power consumption of *CARPO* is closest to that of *Near-optimal*, which assumes the perfect knowledge of future workloads to provide an upper bound.

Figure 8 shows the energy savings for the entire duration of the Wikipedia trace (i.e., seven days) of different power management solutions, compared with the energy consumption of the DCN with no power management. We can see that *CARPO-C* outperforms *ElasticTree* and *GoogleP* on the energy saving performance by 15.7% and 88.7% on average, respecitively. Integrated with link rate adaptation, *CARPO* performs even better than *CARPO-C*. Specifically, *CARPO* outperforms the two baslines by 19.6% and 95%, respectively. After we run the entire workload, we also collect the average delay of all the packets transmitted in the DCN. *CARPO-C* has an average packet delay of 163.8μs with a standard deviation of 47.6μs, while the average packet delay of *ElasticTree* is 137.1μs with a standard deviation of 28.5μs. Note that we can measure the delay only for *CARPO-C*, because the switch used in our testbed does not support dynamic link rate adaptation.

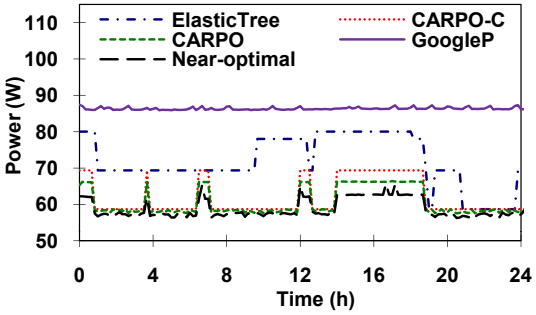


Fig. 7. Power consumption of CARPO and the baselines on testbed running the 4 flows in Figure 1.

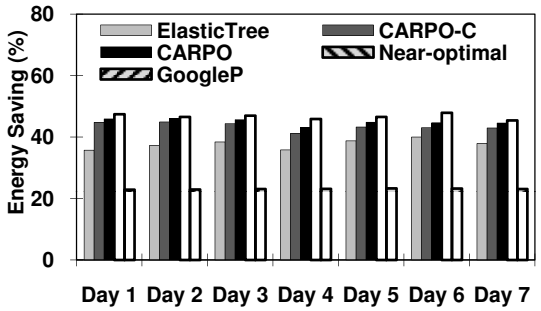


Fig. 8. Energy savings (compared to no power management) on the testbed running the 4 flows in Figure 1.

To stress test CARPO with workloads different from the four flows in Figure 1, we also compose 5 more sets of flows, each with 4 flows randomly selected from the 61 flows in the Wikipedia traces to run on the hardware testbed. Figure 9 shows the energy savings of *CARPO* and the baselines for all the 6 sets of flows. *CARPO* consistently achieves the

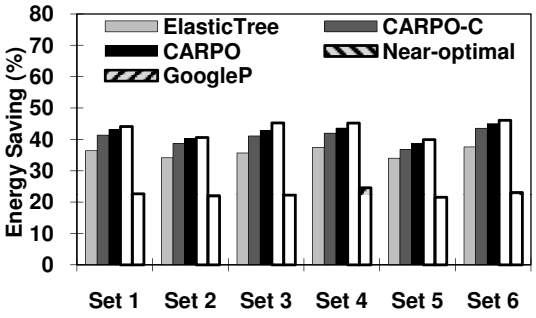


Fig. 9. Energy savings (compared to no power management) on the testbed running randomly selected flow sets.

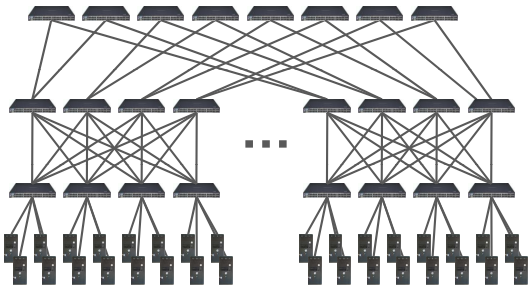


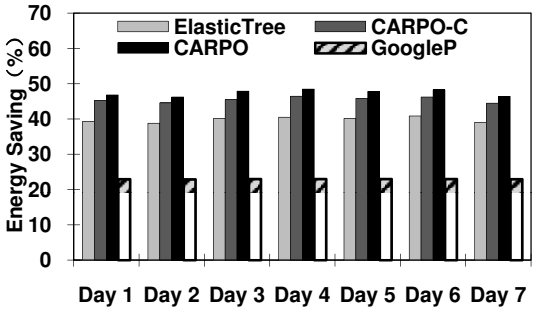
Fig. 10. The *pod-8* fat tree network topology used in simulations.

highest energy savings among all the power management schemes for all the different flow sets. On average, *CARPO* outperforms *ElasticTree* and *GoogleP* on the energy saving performance by 17.7% and 86.3%, respectively.

6 Simulation Evaluation

In this section, we present the simulation results in a larger network topology with all the traffic flows from the Wikipedia traces in OPNET. We use the *fat tree* structure [12] to construct the DCN in our simulation mainly for the purpose of comparison with *ElasticTree*. It is important to note that CARPO can be applied to other network topologies as well. A *pod-k* fat tree structure is composed of k core switches, $\frac{k^2}{2}$ aggregation switches, and $\frac{k^2}{2}$ edge switches. For example, a *pod-8* fat tree structure can hold at most 128 end hosts. If each flow needs to use a pair of dedicated servers, the *pod-8* structure can hold 64 traffic flows. The entire Wikipedia traces consist of 61 folder-grouped flows. Therefore, we use the *pod-8* fat tree topology in our simulation, as shown in Figure 10.

Figure 6 shows the energy savings of *CARPO* and the baselines in this simulation. We cannot present *Near-optimal* here due to the high computational complexity when using the optimization tool with this large number of flows. Similar to the testbed results, *CARPO* has the highest energy saving among all the power management schemes. More specifically, *CARPO* outperforms *ElasticTree* and *GoogleP* on the energy saving performance by 19.1% and 96.3%, respectively. Compared with *CARPO-C* without link rate adaptation, *CARPO* has a 5% more energy saving. The average packet delays of



captionEnergy savings of CARPO and the baselines (compared to no power management) in simulations running the Wikipedia traces.

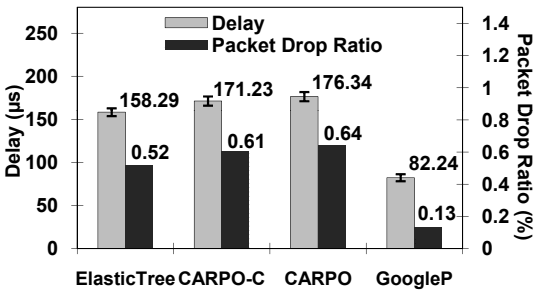


Fig. 11. Average packet delays and packet drop ratios of CARPO and the baselines in simulations running the Wikipedia traces.

all the schemes are presented in Figure 11. Compared with *ElasticTree*, *CARPO* slightly increases the average packet delay only by around 18.05 μs with an increase in the packet drop ratio by 0.12%. These increases are sufficiently small because the delays of all the schemes are much shorter than the typical round-trip time required for the common services in data centers (e.g., $1ms$ as specified in [20]). The slightly increased delay of CARPO comes from its more aggressive consolidation based on the 90-percentile values of the peak bandwidth demands, which may lead to slightly more link capacity violations if workload variations are large. This set of simulations demonstrates that CARPO can effectively save more energy in DCNs, with only negligible increases in the packet delay.

More simulation results on a large-scale DCN and the impacts of different number of sample points used in correlation computation are available in a tech report [21].

7 Conclusion

In this paper, we have presented CARPO, a correlation-aware power optimization scheme for DCNs. CARPO dynamically consolidates traffic flows onto a small set of links and switches in a DCN and then shuts down unused network devices for energy savings. In sharp contrast to existing work, CARPO is designed based on a key observation from the analysis of data center traffic traces that the bandwidth demands of different flows do not peak at exactly the same time. As a result, if the correlations among flows are considered in consolidation, more energy savings can be achieved. CARPO also integrates correlation-aware traffic consolidation with link rate adaptation for maximized energy

savings. We implement CARPO both on a hardware testbed and in simulation. Our empirical results on the hardware testbed running Wikipedia data center traffic workloads demonstrate that CARPO can save up to 46% of network energy for a DCN, while having only negligible delay increases. CARPO also outperforms two state-of-the-art baselines, ElasticTree [3] and GoogleP [4], by 19.6% and 95% on energy savings, respectively. Our simulation results also show the superior energy efficiency of CARPO over the baselines.

References

1. United States Environmental Protection Agency, "Report to congress on server and data center energy efficiency," 2007.
2. A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *SIGCOMM CCR*, 2008.
3. B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "ElasticTree: Saving energy in data center networks," in *NSDI*, 2010.
4. D. Abts, M. Marty, P. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *ISCA*, 2010.
5. P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "Energy aware network operations," in *IEEE Global Internet Symposium*, 2009.
6. M. Gupta, S. Grover, and S. Singh, "A feasibility study for power management in lan switches," in *ICNP*, 2004.
7. S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in *NSDI*, 2008.
8. A. Verma, G. Dasgupta, T. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *USENIX*, 2009.
9. M. Gupta and S. Singh, "Greening of the Internet," in *SIGCOMM*, 2003.
10. G. Ananthanarayanan and R. H. Katz, "Greening the switch," in *HotPower*, 2008.
11. C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "DCell: a scalable and fault-tolerant network structure for data centers," in *SIGCOMM*, 2008.
12. M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *SIGCOMM*, 2008.
13. C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: a high performance, server-centric network architecture for modular data centers," in *SIGCOMM*, 2009.
14. C. Gunaratne, K. Christensen, B. Nordman, and S. Suen, "Reducing the energy consumption of Ethernet with adaptive link rate (ALR)," *Computers, IEEE Transactions on*, 2008.
15. P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in *IFIP Networking*, 2009.

16. T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," in *SIGCOMM*, 2009.
17. Y. Berstein and S. Onn, "The graver complexity of integer programming," *Annals of Combinatorics*, 2009.
18. G. Urdaneta, G. Pierre, and M. van Steen, "Wikipedia workload analysis for decentralized hosting," *Elsevier Computer Networks*, 2009.
19. J. Voß, "Measuring wikipedia," in *10th International Conference of the International Society for Scientometrics and Informetrics*, 2005.
20. V. Vasudevan, A. Phanishayee, H. Shah, E. Krevat, D. G. Andersen, G. R. Ganger, G. A. Gibson, B. Mueller, and P. Inc, "Safe and effective fine-grained TCP retransmissions for datacenter communication," in *SIGCOMM*, 2009.
21. X. Wang, Y. Yao, X. Wang, Q. Cao, and K. Lu, "Correlation-aware power optimization in data center networks," Tech. Rep., 2011. [Online]. Available: http://pacs.ece.utk.edu/trcarpo.pdf